Published in Image Processing On Line on 2023–01–27.
Submitted on 2023–01–10, accepted on 2023–01–10.
ISSN 2105–1232 © 2023 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2023.459

# Monocular Depth Estimation: a Review of the 2022 State of the Art

Thibaud Ehret

Université Paris-Saclay, ENS Paris-Saclay, Centre Borelli, Gif-sur-Yvette, France
thibaud.ehret@ens-paris-saclay.fr

*Communicated by* Jean-Michel Morel *Demo edited by* Thibaud Ehret

## Abstract

We compare five monocular depth estimation methods based on deep learning. This comparison focuses on how well methods generalize rather than a quantitative comparison on a specific dataset. This study shows that while monocular depth estimation methods work well on images similar to training images, they often show artifacts when applied on images out of the training distribution. We evaluate the different methods with images similar to training data and images with unusual point of views (e.g. top-down) or paintings. The readers are invited to judge by themselves about the advantages and drawbacks of all methods by submitting their own images to the online demo associated with the present paper.

## Source Code

The source codes and documentation for the algorithms presented in this paper are available from the web page of this article[1]. Usage instructions are included in the `README.md` file of each archive. The original implementations of the methods are available at the following links: MiDaS and DPT methods[2], Adabins method[3], GLPDepth method[4], 3DShape method[5].
This is an MLBriefs article, the source codes have not been reviewed!

**Keywords:** depth; monocular depth estimation; deep learning; comparison; cutdepth; adabins; 3d; midas; dpt

[1] https://doi.org/10.5201/ipol.2023.459
[2] https://github.com/isl-org/MiDaS
[3] https://github.com/shariqfarooq123/AdaBins
[4] https://github.com/vinvino02/GLPDepth
[5] https://github.com/aim-uofa/AdelaiDepth

# 1 Introduction

The goal of monocular depth estimation is to predict the distance to the scene using only a single RGB image as input. This is particularly important in setups where other 3D acquisition modes such as LiDaR or stereo vision, are not possible or too costly such as automatic driving [23] or robotic [18]. It can also be useful to generate realistic training data for other applications, such as optical flow [30]. In this paper, we compare five recent deep-learning based monocular depth estimation methods: GPLDepth [15] (2022), Adabins [1] (2021), 3DShape [34] (2021), MiDaS [22] (2020) and DPT [21] (2021). The authors of all these methods made their code and pretrained network freely available allowing to reproduce their results. In Sections 2 to 6, we briefly describe the five methods compared in this paper. In Section 7 we compare visually the results of the methods. The comparison is performed using the inferno colormap (see Figure 1). All these methods are parameters free allowing for a simpler comparison between all methods.

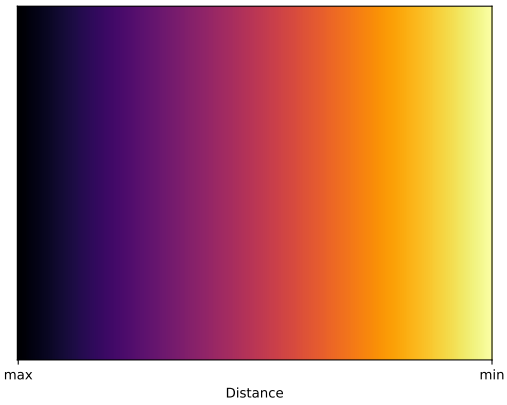


Figure 1: Colormap used for the comparison. Dark purple corresponds to the furthest distances and bright yellow to the closest distances.

# 2 GLPDepth [15]

GLPDepth uses a transformer based encoder-decoder (see Figure 2) to predict the pixel wise depth from the embedded patches of the input image. The main improvement over previous methods is the use of a newly proposed data augmentation trick called *vertical cutdepth*. This is an extension of the data augmentation method presented in [14]. The main idea of [14] is to replace parts of the input image by the ground truth depth. While [14] proposes to select the replaced regions completely at random, *vertical cutdepth* suggests that it is better to replace vertical bands of the image since monocular depth estimation methods use mainly the vertical information for the prediction [6]. An example of augmentation is shown in Figure 3.

GLPDepth is trained on the *NYU depth V2* dataset using the scale invariant log-scale loss [7]

$$L = \frac{1}{n}\sum_i (\log(d_i) - \log(\hat{d}_i))^2 - \frac{1}{2n^2}\left(\sum_i (\log(d_i) - \log(\hat{d}_i))\right)^2, \tag{1}$$

with $n$ the number of pixels in the batch, $d$ the ground truth depth and $\hat{d}$ the depth predicted by the neural network.

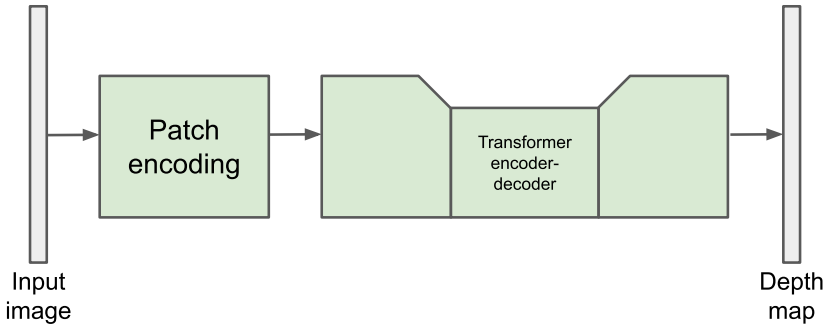


Figure 2: Architecture of GLPDepth [15].

Figure 3: From top left to bottom right: reference image, ground truth depth (black corresponds to unknown data), augmentation as proposed in [14] and augmentation as proposed in [15].

# 3 Adabins [1]

Instead of predicting directly the distance to the scene, Adabins considers the problem of depth estimation as a classification problem as proposed in [9]. The idea is that it is easier to predict a bin rather than directly the depth. The main difference with [9] is how the bins are computed. In the original work, each bin had a predetermined width while in the proposed work this width is adaptive and depends on the image.

Adabins is based on an encoder-decoder architecture using EfficientNet B5 [28] as a backbone. It is then followed by the proposed adaptive bin estimator called *Adabin* based on a transformer that predicts both the bin widths and the probability of each pixel to belong to a given bin (see

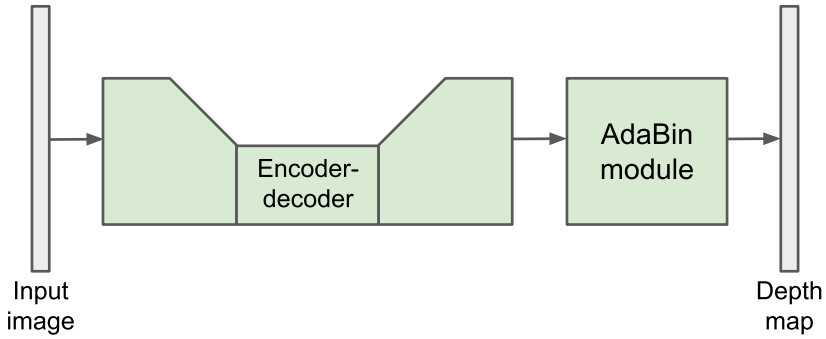


Figure 4: Architecture of Adabins [1].

Figure 4). Instead of returning the center of the most likely bin (thus creating a quantized output), the expected depth is instead returned. The expected depth corresponds to

$$\hat{d} = \sum_{k=1}^{K} c(\mathbf{b}_k) p_k, \tag{2}$$

with $K$ the number of bins, $c(\mathbf{b})$ the center of bin $\mathbf{b}$ and $p_k$ the output probability that the pixel should be located in bin $\mathbf{b}_k$.

The network is trained on either *NYU depth V2* [25], *KITTI* [10] or SUN-RGBD [26]. The loss used is a combination of two losses:

- The scale invariant log-scale loss [7]

$$L_{pixel} = \alpha \sqrt{\frac{1}{n}\sum_i (\log(d_i) - \log(\hat{d}_i))^2 - \frac{\lambda}{2n^2}\left(\sum_i (\log(d_i) - \log(\hat{d}_i))\right)^2}, \tag{3}$$

  with $n$ the number of pixels in the batch, $\lambda = 0.85$ and $\alpha = 10$, $d$ the ground truth depth and $\hat{d}$ the depth predicted by the neural network;

- The bin center density loss that encourages the distribution of the predicted bin centers to follow the distribution of the ground truth. This is done using a bi-directional Chamfer distance [8]

$$L_{bins} = \sum_i \min_j \|d_i - c(\mathbf{b}_j)\|_2^2 + \sum_j \min_i \|d_i - c(\mathbf{b}_j)\|_2^2, \tag{4}$$

  with $c(\mathbf{b})$ corresponding to the center of bin $\mathbf{b}$.

The final loss is then $L = L_{pixel} + 0.1 L_{bins}$.

# 4 MiDaS [22]

Contrary to the previous methods, MiDaS does not propose a new architecture or a new loss but instead shows that combining multiple training datasets enables better performance and better generalization. In fact, most of the work focuses on uniformizing the training and test pipelines across the different datasets (DIML [16], MegaDepth [17], ReDWeb [31], WSVD [29], 3D movies, DIW [3], ETH3D [24], Sintel [2], KITTI [10, 19], NYU Depth V2 [25], TUM-RGBD [27]).

They also compare different encoder-decoder architectures based on either ResNet-50 [11], ResNeXt-101 [33] or DenseNet-161 [13] (see Figure 7 for the architecture). The authors observed that using a base network that performs better on a classification task (such as Imagenet [5]) leads to better monocular depth estimation performance.

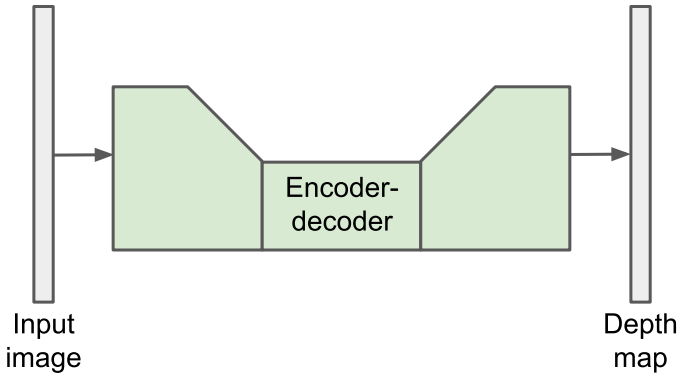


Figure 5: Architecture of MiDaS [22].

# 5 DPT [21]

DPT extends MiDaS by changing the base architecture to a transformer based encoder-decoder architecture (see Figure 6).

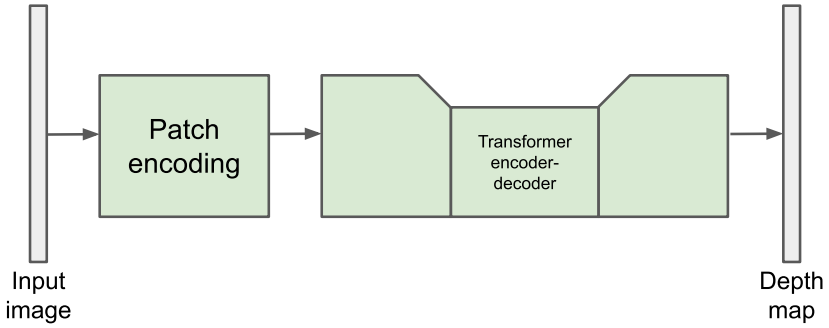


Figure 6: Architecture of DPT [21].

# 6 3DShape [34]

3DShape also studies the problem of creating a 3D model from the estimated depth map. This part is, however, not of interest for this review. The 3D model predicted is nonetheless used during training to regularize the monocular depth prediction module using the surface normals of the estimated 3D shape. Therefore, three different losses are combined during training:

- An image-level normalized regression loss

$$L_{ILNR} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{d}_i - d_i^*\right| + \left|\tanh(\hat{d}_i/100) - \tanh(d_i^*/100)\right|, \tag{5}$$

  with $\hat{d}$ the depth predicted by the network and $d^*$ the mean and variance normalized ground truth (10% of outliers are discarded during the estimation of these two parameters).

- A pairwise normal loss similar to [32]

$$L_{PWN} = \frac{1}{N}\sum_{i=1}^{N}\left|n_{A_i}.n_{B_i} - n^*_{A_i}.n^*_{B_i}\right|, \tag{6}$$

with $A$ and $B$ two sets of paired points sampled on edges and planes of the 3D structure, $n$ (respectively $n^*$) is the normal estimated on the predicted 3D point cloud (respectively ground truth point cloud).

- A multiscale gradient loss similar to [17]

$$L_{MSG} = \frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N}\left|\nabla_x \hat{d}_i^k - \nabla_x d_i^{k,*}\right| + \left|\nabla_y \hat{d}_i^k - \nabla_y d_i^{k,*}\right|, \tag{7}$$

with $d^k$ the depth at scale $k$.

The final loss is then $L = L_{ILNR} + L_{PWN} + 0.5L_{MSG}$.

The architecture of 3DShape is an encoder-decoder with ResNeXt-101 [33] as backbone (see Figure 7). Similarly to MiDaS [22] or DPT [21], the network is trained using multiple datasets at the same time (Taskonomy [35], 3D Ken Burns [20], DIML [16], Holopix50K [12] and HRWSI [32]).

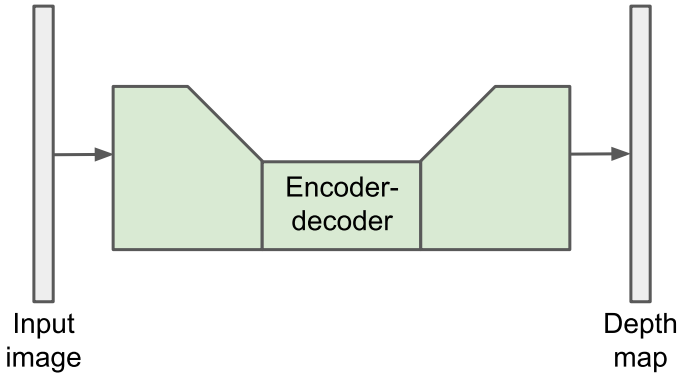


Figure 7: Architecture of 3DShape [34].

# 7 Comparative Experiments

The different experiments can be classified into three main categories: natural images similar to the training data, synthetic or clearly-non natural images with depth cues and, finally, voluntarily ambiguous images such as *trompe l'œil*. The goal is to drive all methods into a corner and try to find out what their limit and generalization capabilities are.

**Natural images.** The first set of experiments consists in testing natural images similar to the training images. We start by comparing the results on a photo of a bookstore from the *NYU depth V2* dataset [25] (Figure 8), which is one of the training datasets used by monocular depth estimation methods, with a similar photo of classroom that is not part of the training dataset (Figure 9). As expected, all methods perform well on these examples. There are nonetheless several major differences between the methods. GPLDepth [15], Adabins [1] and 3DShape have a much deeper field of view than MiDaS [22] and DPT [21]. Indeed, far away objects (such as the bookcase on the opposite wall in Figure 8) are visible in the first three methods but not in the last two. Moreover, Adabins [1] seems to have some ghosting (or transparency effects). The center of a solid object is sometimes shown as further away than the boundaries of the same object.

We then look at two natural images that are out of the training dataset distribution. The first, Figure 10, has a large reflective structure in the middle of it and the other, Figure 11, is a photo

from inside a cave and looking out. We can already see that some methods, namely GPLDeth [15] and Adabins [1], start breaking down. The other methods produce reasonable results even though DPT [21] seems to struggle a bit with the reflective structure in Figure 10. However, DPT [21] produces the most precise cutout of the ceiling of the cave in Figure 11.

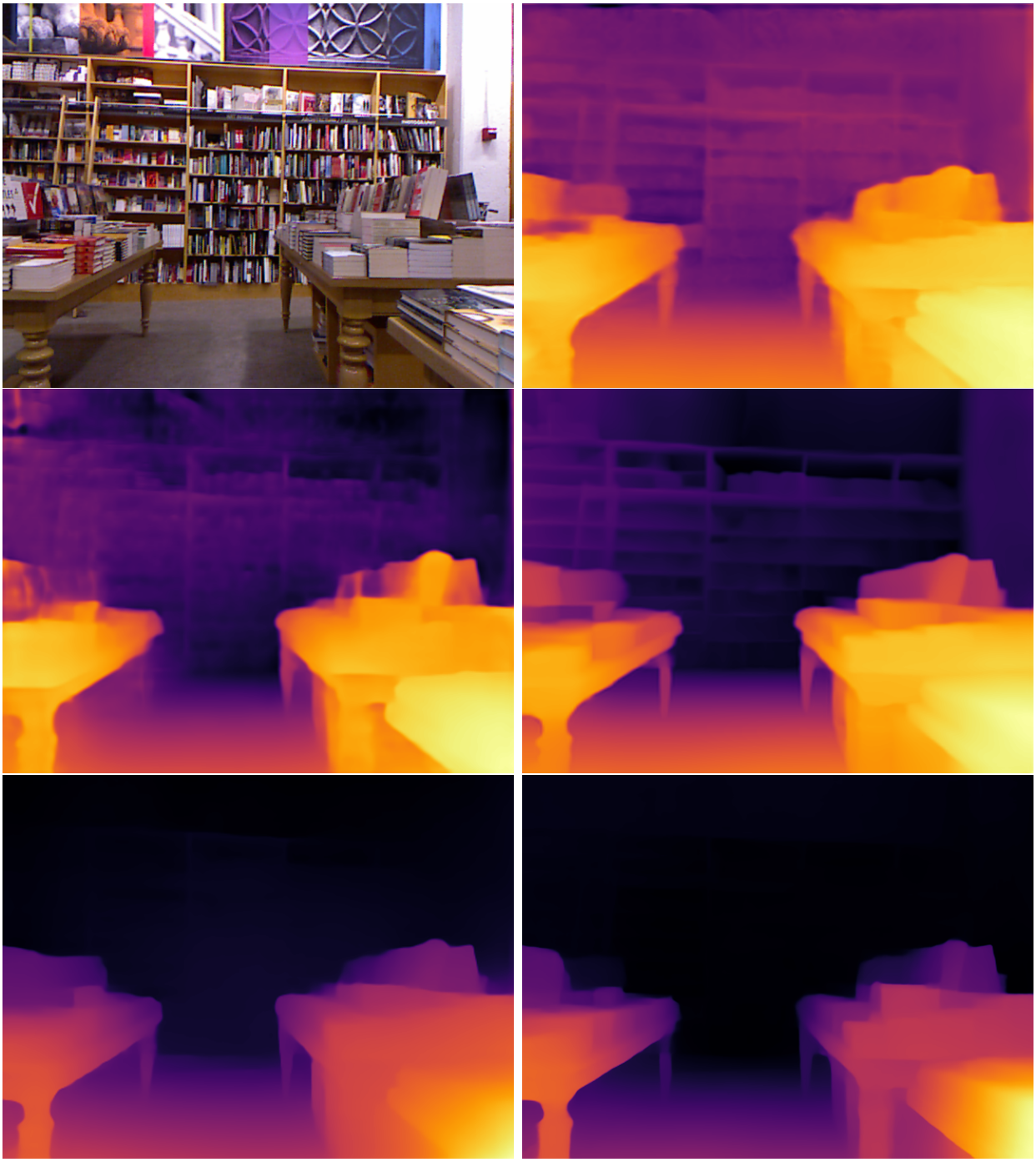

Figure 8: Comparison of five monocular depth estimation methods on a bookstore image from the *NYU depth V2* dataset [25]: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

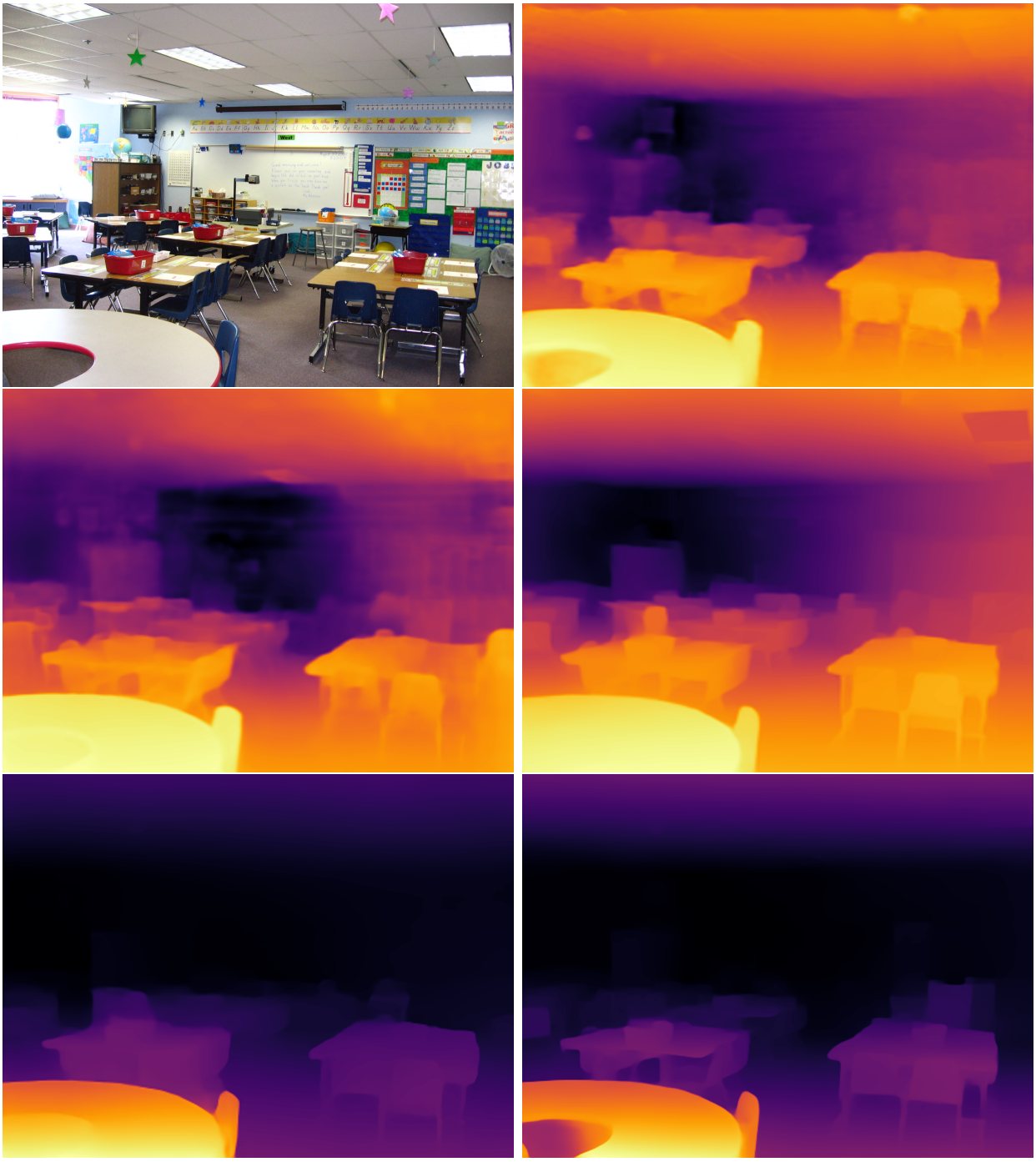

Figure 9: Comparison of five monocular depth estimation methods on a classroom photo similar to the training data: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

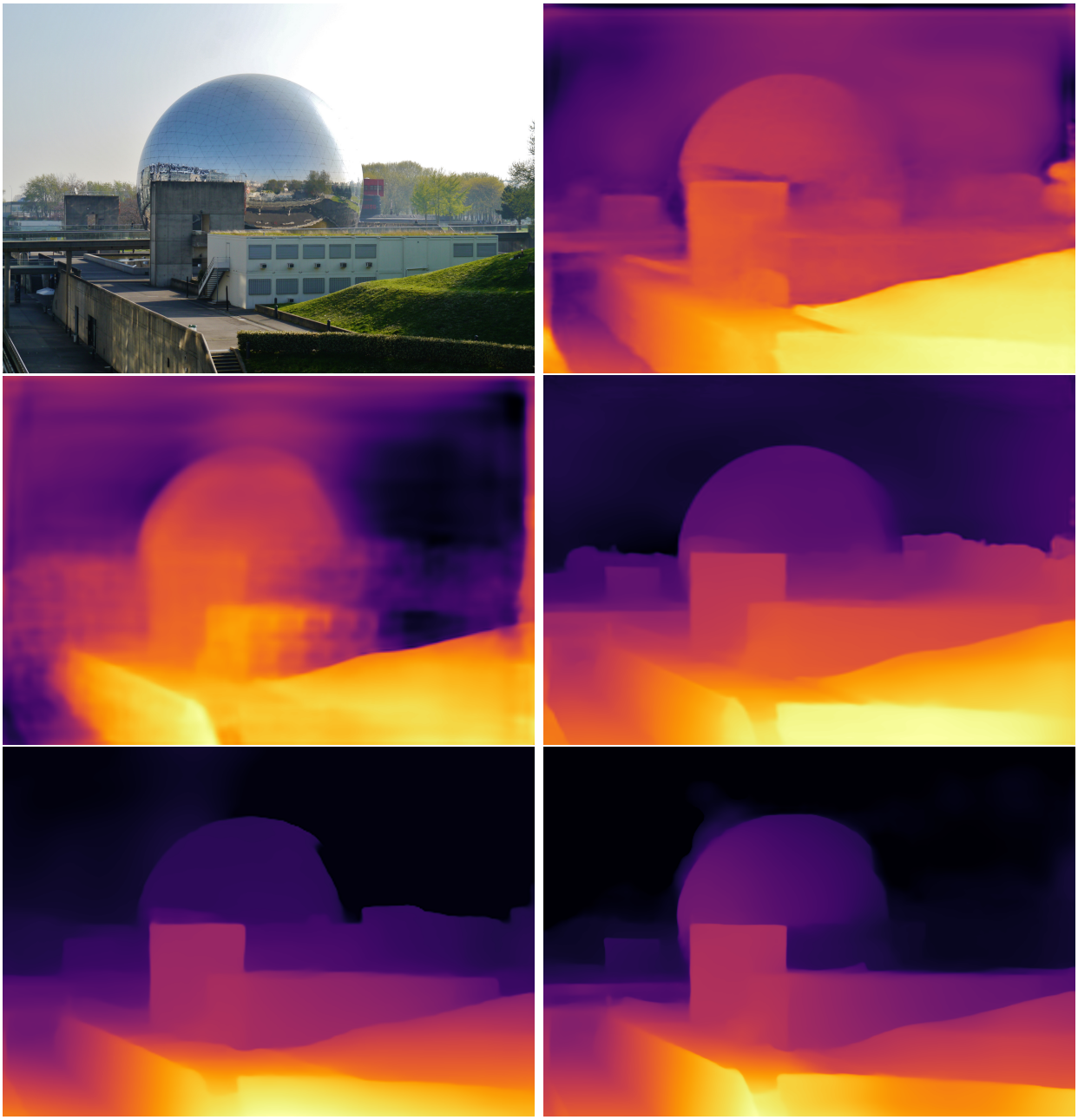

Figure 10: Comparison of five monocular depth estimation methods on a photo of *La Geode* in parc de la Villette, Paris: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

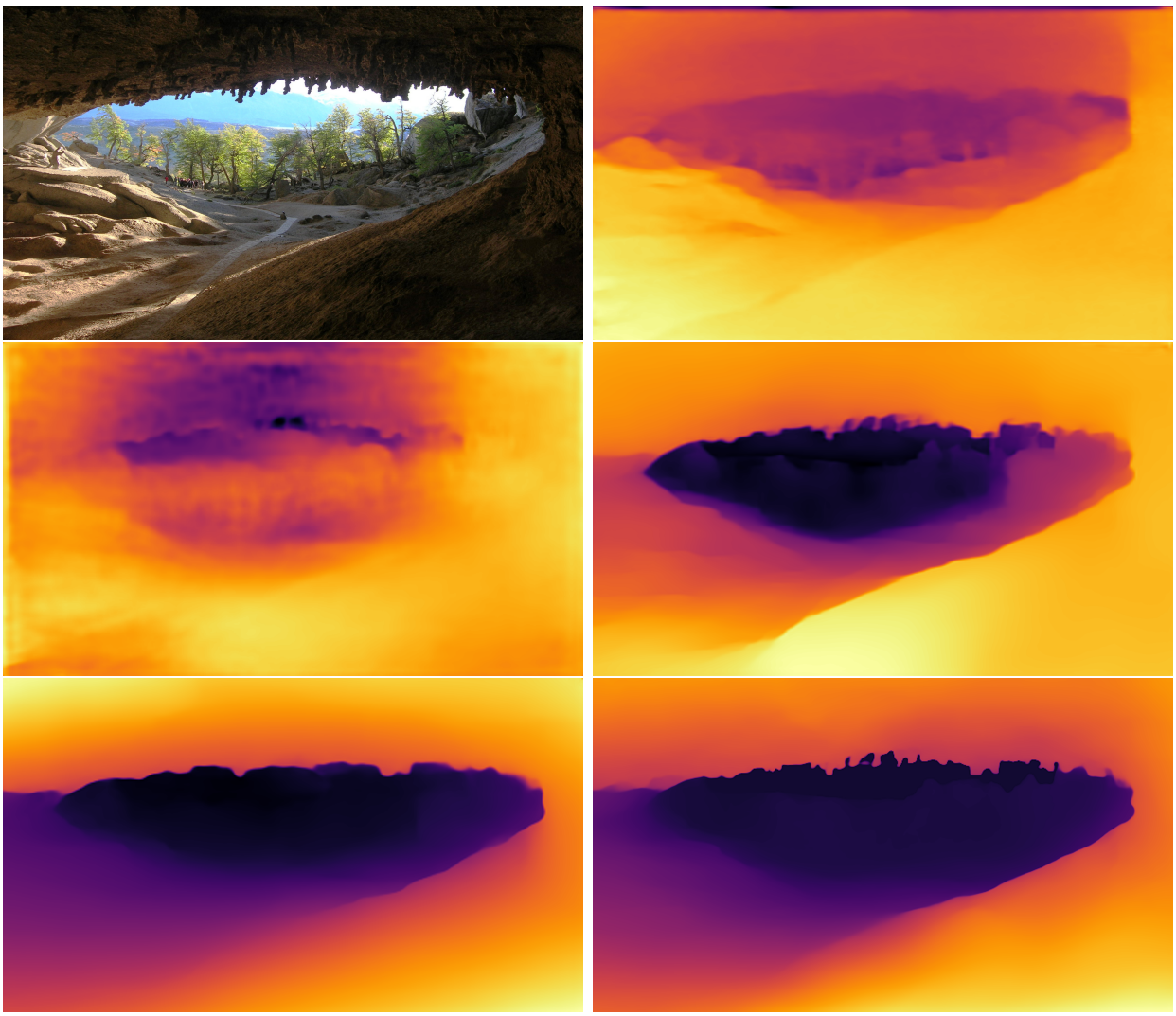

Figure 11: Comparison of five monocular depth estimation methods on an inside cave photo: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

We finally test the methods on an overhead view of a city (Figure 12). While the image is natural, it has a much different point of view than the regular training data. Only DPT [21] seems to produce reasonable results. 3DShape [34] and MiDaS [22] seem to present a bias: objects near the bottom of the image should be closer than those at the top. This is likely because most natural outdoor images verify this bias (ground on the bottom and sky on the top).

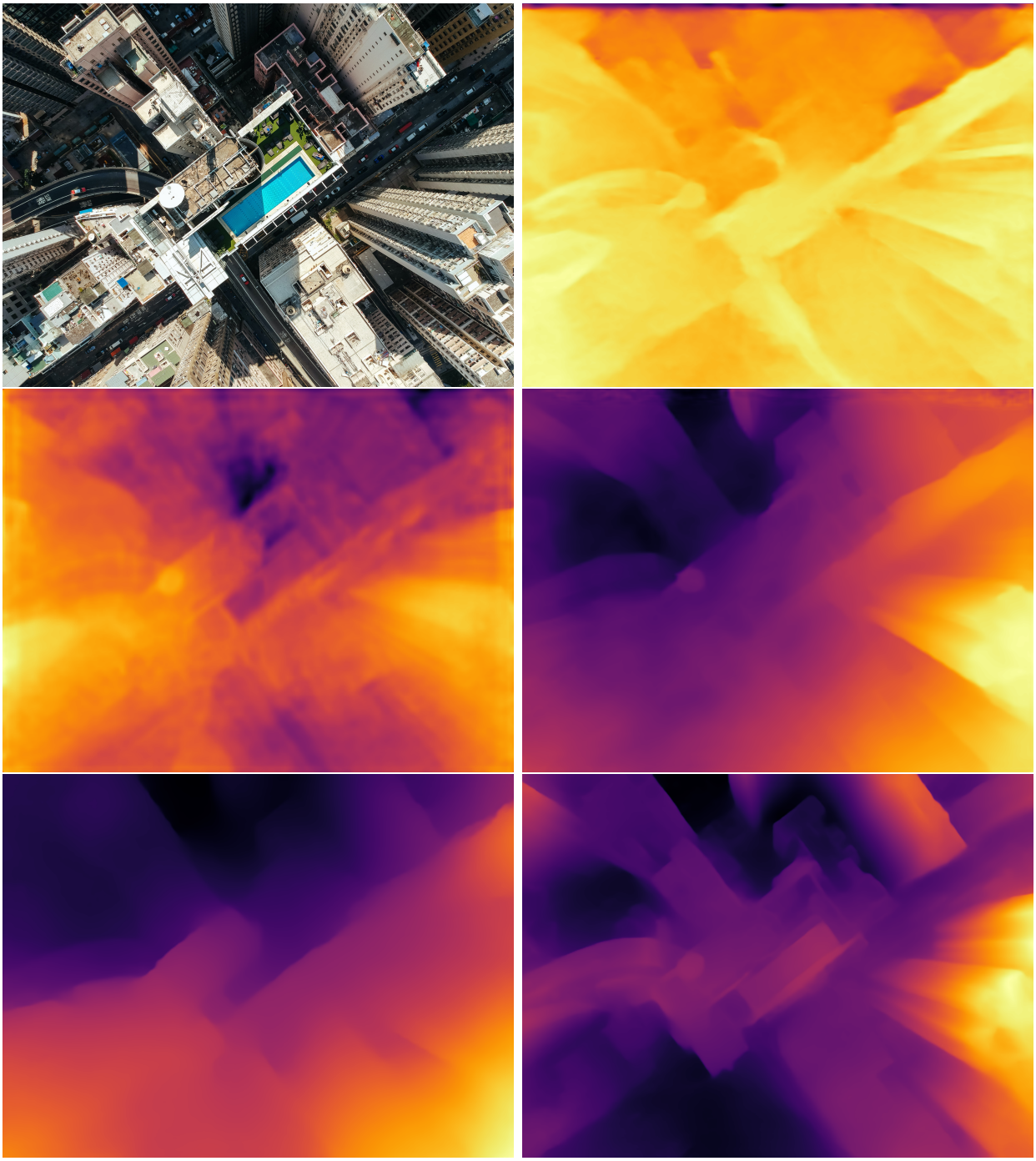

Figure 12: Comparison of five monocular depth estimation methods on an overhead view photo: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

**Synthetic and non-natural images.** The second set of experiments is performed on non-natural images. The first example is a synthetic image created by stacking rectangles on top of one another (Figure 13). The relative distance to each rectangle can nonetheless be estimated using the relative ordering and cues such as T-junctions (see [4] for more details). On this example, all methods perform badly. Most of the time, the distance to a given rectangle is not constant and the borders are not well defined. We can also observe a similar bias to the one presented for Figure 12.

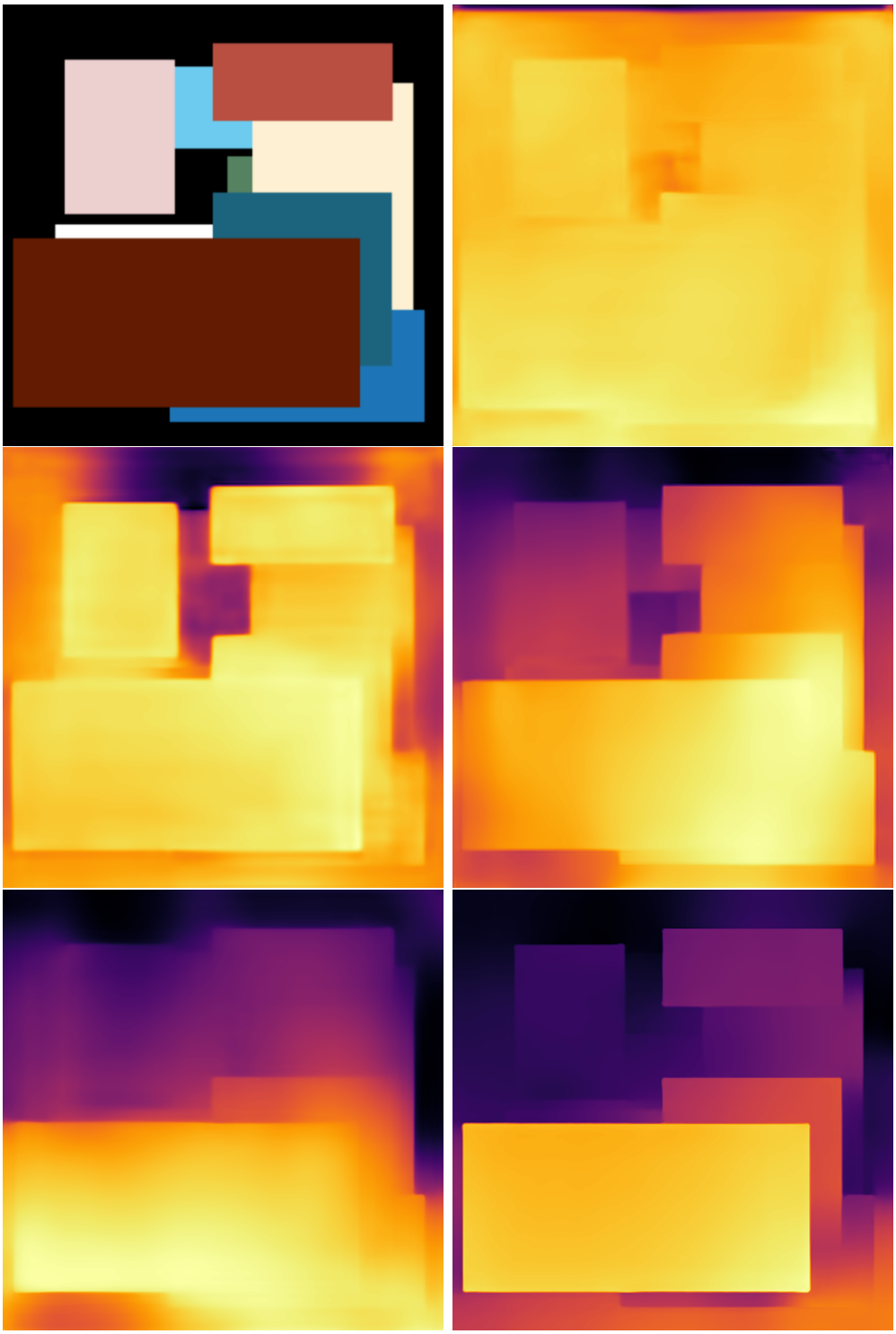

Figure 13: Comparison of five monocular depth estimation methods on a synthetic image showing stacks of rectangles: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

The second example is using a famous abstract painting, *Starry Night* by Van Gogh (Figure 14). Even though this example is completely different than the training data, some methods, such as 3DShape [34], MiDaS [22] and DPT [21], produce reasonable results: the large structure is said to be in front of the scene and the bottom, with the village, closer than the sky.

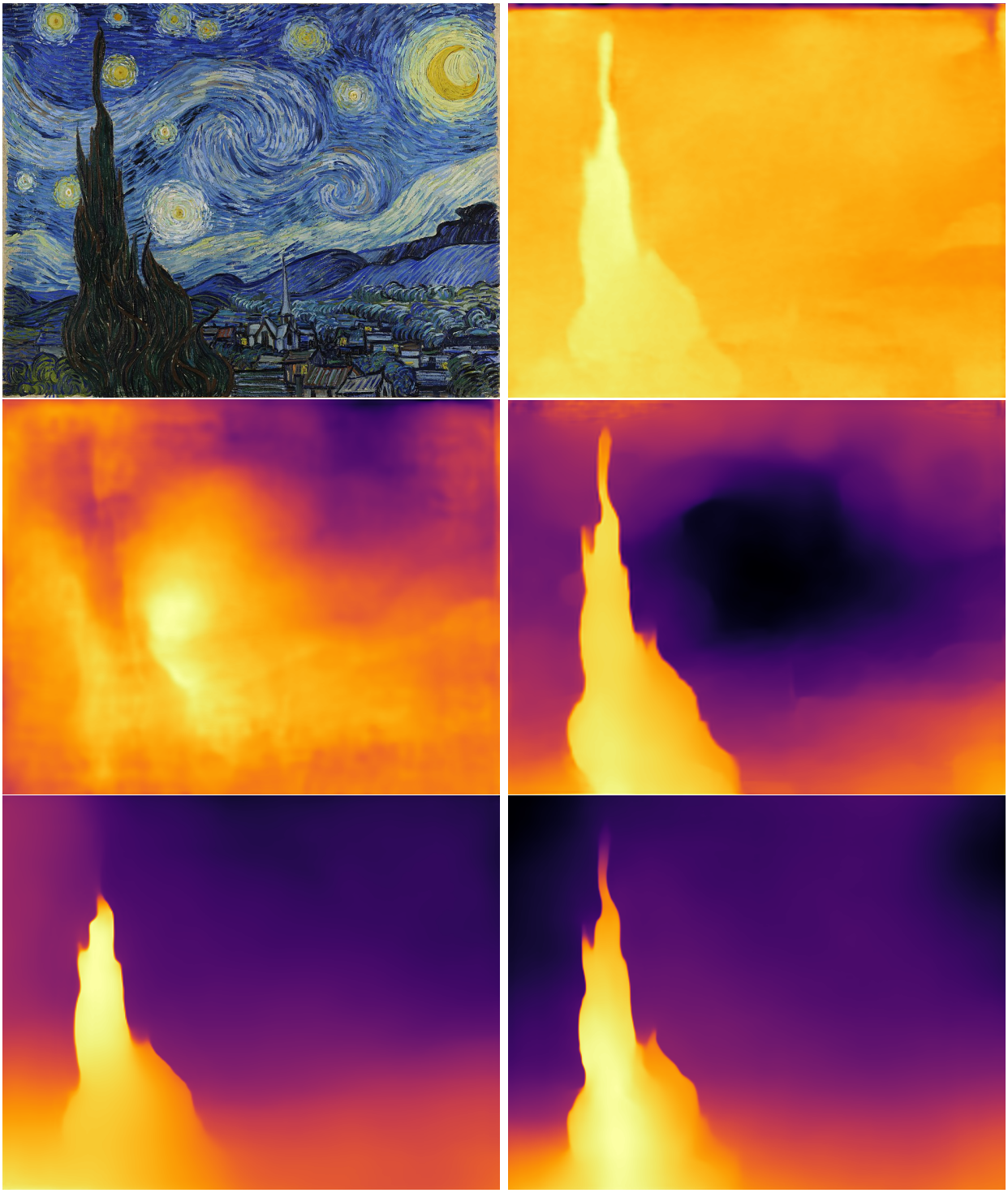

Figure 14: Comparison of five monocular depth estimation methods on the *Starry Night* painting by Van Gogh: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

**Ambiguous images.** The last set of experiments is performed on voluntarily ambiguous images, namely *trompe l'œil*. Therefore there is no right or wrong answer since the goal of these images is to find out whether the different methods can also be tricked into seeing a different depth than in reality. The first example is a park in Paris that was made to appear as a sphere when looked from a specific point of view (Figure 15). While most methods see a sphere in the location of the park, DPT [21] seems to see a flat structure that rises above the ground.

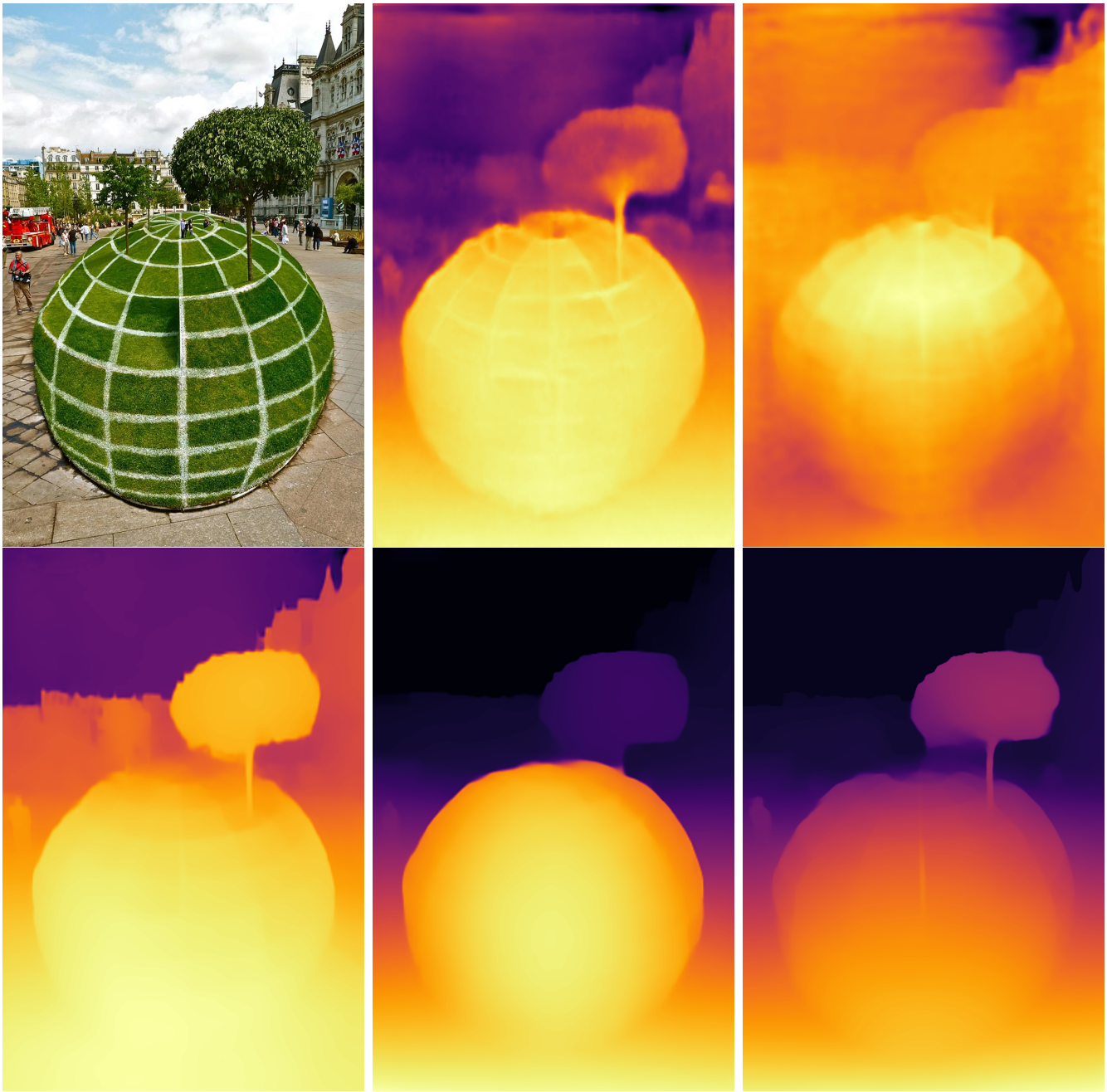

Figure 15: Comparison of five monocular depth estimation methods on a photo of a park in Paris: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

The second example is of a hallway with a tiling that makes it appear not flat (Figure 16). Surprisingly, most methods see the hallway as flat and are not fooled by this visual trick. A slight change of depth is nonetheless visible for GPLDepth [15] and Adabins [1].

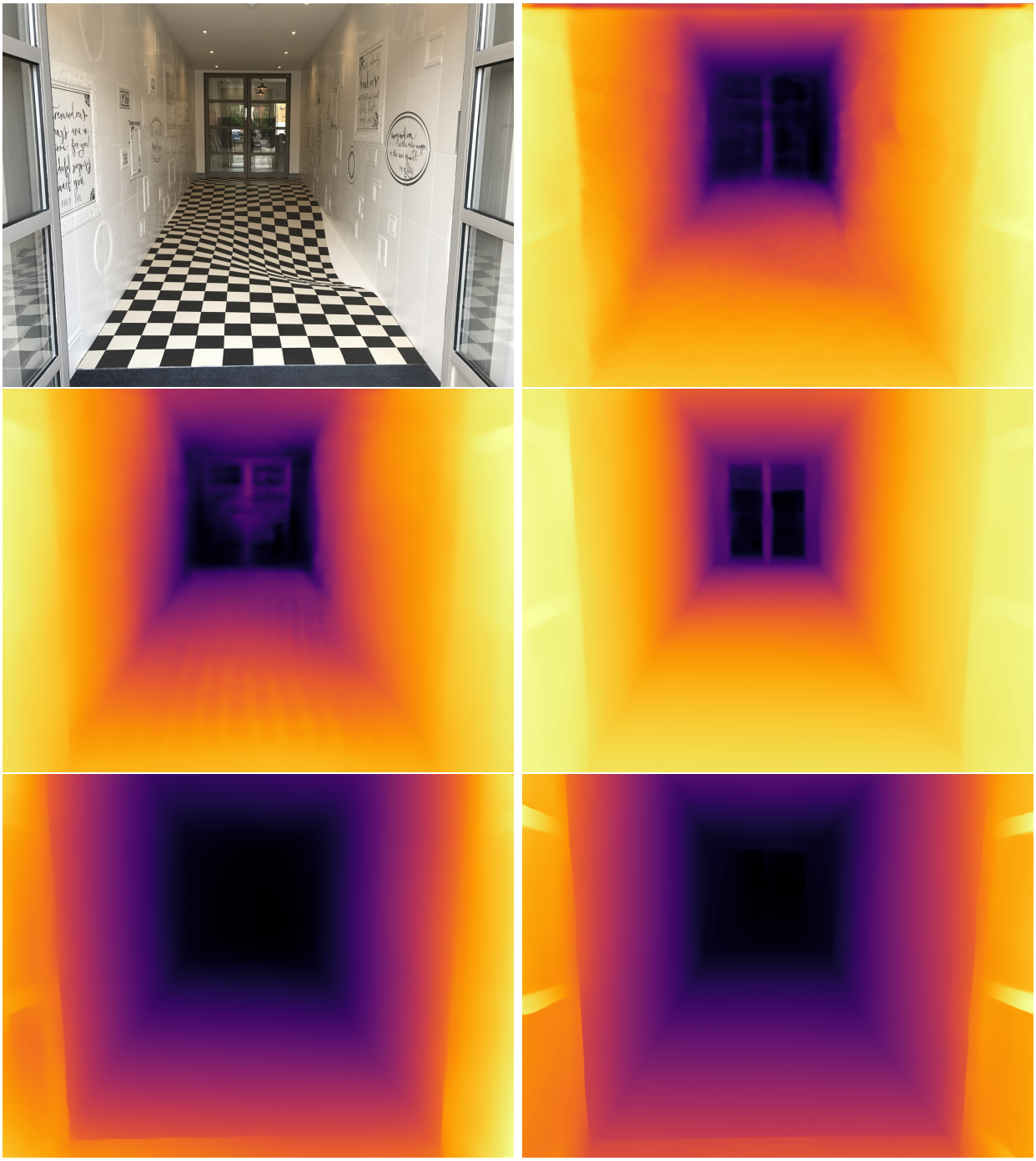

Figure 16: Comparison of five monocular depth estimation methods on a photo of a hallway: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

The last example is a photo of a wall with a realistic painting creating a depth illusion (Figure 17). In this case, all methods are tricked by the realistic painting and predict a hallway where only a wall exists.

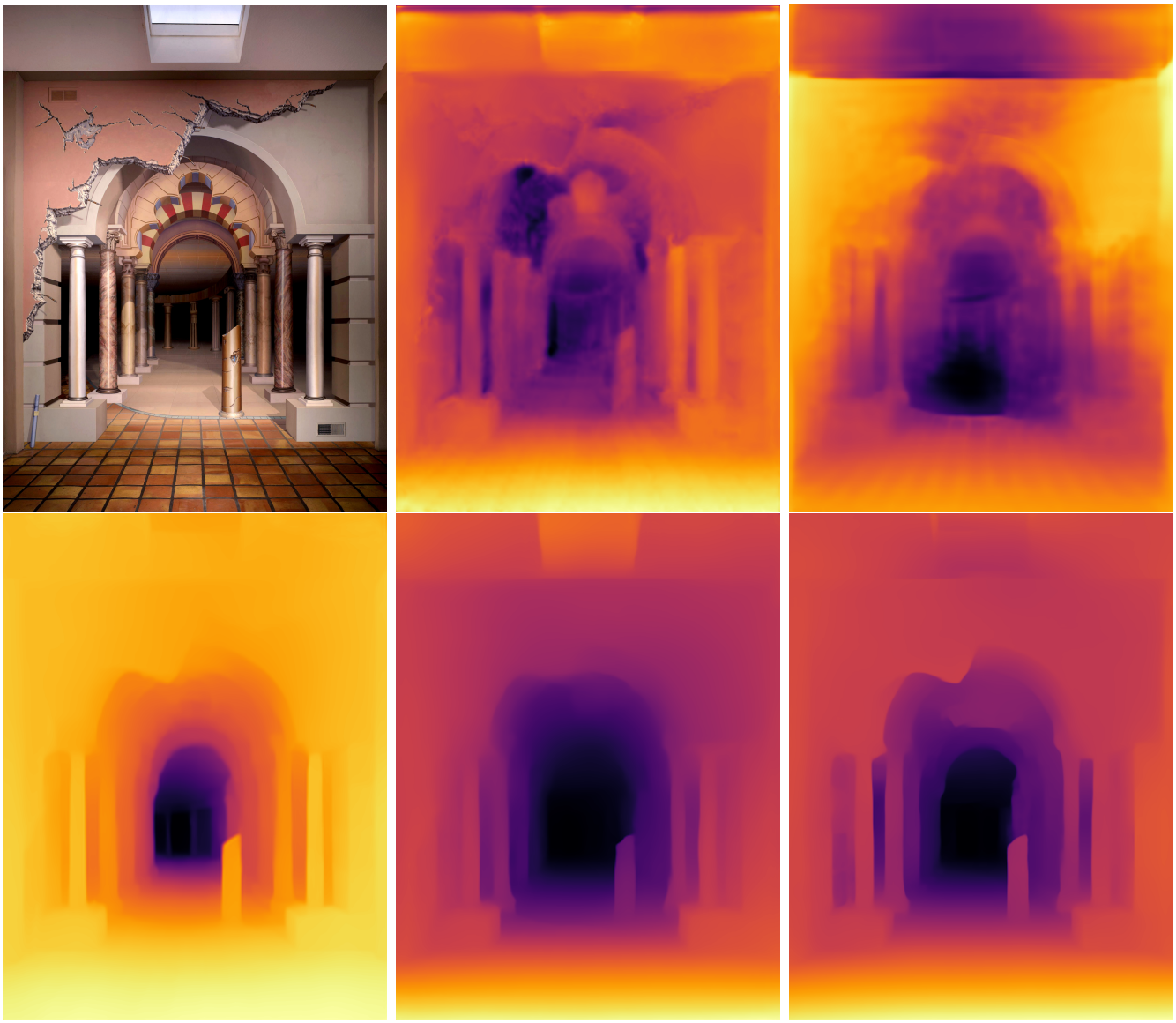

Figure 17: Comparison of five monocular depth estimation methods on a photo of a wall: the input, results of GPLDepth [15], Adabins [1], 3DShape [34], MiDaS [22], DPT [21].

**Computation time.** We also compare the computation times for the different methods on a Intel(R) Core(TM) i7-7820HQ CPU in Table 1.

Table 1: Computation time for the different methods presented. Computation times (in seconds) were estimated using the image shown in Figure 9.

| GPLDepth [15] | Adabins [1] | 3DShape [34] | MiDaS [22] | DPT [21] |
|---|---|---|---|---|
| 12.99 | 6.88 | 13.77 | 5.81 | 12.22 |

# Image Credits

 *NYU depth V2* dataset [25]

     Wikipedia

 Robert Bye

 Adrien Courtois et al. [4]

 François Abélanet

 Casa Ceramika

# References


[1] S. F. Bhat, I. Alhashim, and P. Wonka, *Adabins: Depth estimation using adaptive bins*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 4009–4018. https://doi.org/10.1109/CVPR46437.2021.00400.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, *A naturalistic open source movie for optical flow evaluation*, in European Conference on Computer Vision (ECCV), Springer, 2012, pp. 611–625. https://doi.org/10.1007/978-3-642-33783-3_44.

[3] W. Chen, Z. Fu, D. Yang, and J. Deng, *Single-image depth perception in the wild*, Advances in Neural Information Processing Systems, 29 (2016). https://dl.acm.org/doi/10.5555/3157096.3157178.

[4] A. Courtois, J.-M. Morel, and P. Arias, *Investigating neural architectures by synthetic dataset design*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2022, pp. 4890–4899. https://doi.ieeecomputersociety.org/10.1109/CVPRW56347.2022.00536.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A large-scale hierarchical image database*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Ieee, 2009, pp. 248–255. https://doi.org/10.1109/CVPR.2009.5206848.

[6] T. v. Dijk and G. d. Croon, *How do neural networks see depth in single images?*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 2183–2191. https://doi.org/10.1109/ICCV.2019.00227.

[7] D. Eigen, C. Puhrsch, and R. Fergus, *Depth map prediction from a single image using a multi-scale deep network*, Advances in Neural Information Processing Systems, 27 (2014). https://dl.acm.org/doi/10.5555/2969033.2969091.

[8] H. Fan, H. Su, and L. J. Guibas, *A point set generation network for 3D object reconstruction from a single image*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 605–613. https://doi.org/10.1109/CVPR.2017.264.

[9] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, *Deep ordinal regression network for monocular depth estimation*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2002–2011. https://doi.org/10.1109/CVPR.2018.00214.

[10] A. Geiger, P. Lenz, and R. Urtasun, *Are we ready for autonomous driving? The KITTI vision benchmark suite*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2012, pp. 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074.

[11] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

[12] Y. Hua, P. Kohli, P. Uplavikar, A. Ravi, S. Gunaseelan, J. Orozco, and E. Li, *Holopix50k: A Large-Scale In-the-wild Stereo Image Dataset*, in CVPR Workshop on Computer Vision for Augmented and Virtual Reality, 2020.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700–4708. https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.243.

[14] Y. Ishii and T. Yamashita, *CutDepth: Edge-aware Data Augmentation in Depth Estimation*, arXiv preprint arXiv:2107.07684, (2021). https://doi.org/10.48550/arXiv.2107.07684.

[15] D. Kim, W. Ga, P. Ahn, D. Joo, S. Chun, and J. Kim, *Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth*, arXiv preprint arXiv:2201.07436, (2022). https://doi.org/10.48550/arXiv.2201.07436.

[16] Y. Kim, H. Jung, D. Min, and K. Sohn, *Deep monocular depth estimation via integration of global and local predictions*, IEEE Transactions on Image Processing, 27 (2018), pp. 4131–4144. https://doi.org/10.1109/TIP.2018.2836318.

[17] Z. Li and N. Snavely, *MegaDepth: Learning single-view depth prediction from internet photos*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2041–2050. https://doi.org/10.1109/CVPR.2018.00218.

[18] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza, *Toward domain independence for learning-based monocular depth estimation*, IEEE Robotics and Automation Letters, 2 (2017), pp. 1778–1785. https://doi.org/10.1109/LRA.2017.2657002.

[19] M. Menze and A. Geiger, *Object scene flow for autonomous vehicles*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3061–3070. https://doi.org/10.1109/CVPR.2015.7298925.

[20] S. Niklaus, L. Mai, J. Yang, and F. Liu, *3D Ken Burns effect from a single image*, ACM Transactions on Graphics (ToG), 38 (2019), pp. 1–15. https://doi.org/10.1145/3355089.3356528.

[21] R. Ranftl, A. Bochkovskiy, and V. Koltun, *Vision transformers for dense prediction*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 12179–12188. https://doi.org/10.1109/ICCV48922.2021.01196.

[22] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, *Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer*, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2020). https://doi.org/10.1109/TPAMI.2020.3019967.

[23] M. Schön, M. Buchholz, and K. Dietmayer, *MGNet: Monocular geometric scene understanding for autonomous driving*, in IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15804–15815. https://doi.org/10.1109/ICCV48922.2021.01551.

[24] T. Schops, J. L. Schonberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, *A multi-view stereo benchmark with high-resolution images and multi-camera videos*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3260–3269. https://doi.org/10.1109/CVPR.2017.272.

[25] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, *Indoor segmentation and support inference from RGBD images*, in European Conference on Computer Vision (ECCV), Springer, 2012, pp. 746–760. https://doi.org/10.1007/978-3-642-33715-4_54.

[26] S. Song, S. P. Lichtenberg, and J. Xiao, *SUN RGB-D: A RGB-D scene understanding benchmark suite*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 567–576. https://doi.org/10.1109/CVPR.2015.7298655.

[27] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, *A benchmark for the evaluation of RGB-D SLAM systems*, in IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 573–580. https://doi.org/10.1109/IROS.2012.6385773.

[28] M. Tan and Q. Le, *EfficientNet: Rethinking model scaling for convolutional neural networks*, in International Conference on Machine Learning (ICLM), PMLR, 2019, pp. 6105–6114. https://doi.org/10.48550/arXiv.1905.11946.

[29] C. Wang, S. Lucey, F. Perazzi, and O. Wang, *Web stereo video supervision for depth prediction from dynamic scenes*, in International Conference on 3D Vision (3DV), IEEE, 2019, pp. 348–357. https://doi.org/10.1109/3DV.2019.00046.

[30] J. Watson, O. M. Aodha, D. Turmukhambetov, G. J. Brostow, and M. Firman, *Learning stereo from single images*, in European Conference on Computer Vision (ECCV), Springer, 2020, pp. 722–740. https://doi.org/10.1007/978-3-030-58452-8_42.

[31] K. Xian, C. Shen, Z. Cao, H. Lu, Y. Xiao, R. Li, and Z. Luo, *Monocular relative depth perception with web stereo data supervision*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 311–320. https://doi.org/10.1109/CVPR.2018.00040.

[32] K. Xian, J. Zhang, O. Wang, L. Mai, Z. Lin, and Z. Cao, *Structure-guided ranking loss for single image depth prediction*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 611–620. https://doi.org/10.1109/CVPR42600.2020.00069.

[33] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, *Aggregated residual transformations for deep neural networks*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1492–1500. https://doi.org/10.1109/CVPR.2017.634.

[34] W. Yin, J. Zhang, O. Wang, S. Niklaus, L. Mai, S. Chen, and C. Shen, *Learning to recover 3D scene shape from a single image*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 204–213. https://doi.org/10.1109/CVPR46437.2021.00027.

[35] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, *Taskonomy: Disentangling task transfer learning*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 3712–3722. https://doi.org/10.1109/CVPR.2018.00391.